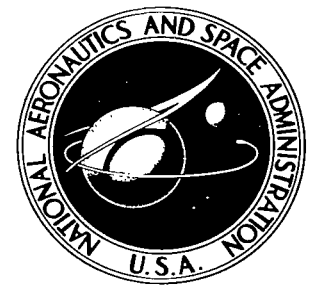
NASA TECHNICAL NOTE

NASA TN D-3027



NASA TN D-3027

ILLUSTRATION OF DESIGN METHODS FOR POWER REDUCTION IN LOGICAL SYSTEMS

by H. Allen Curtis

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • SEPTEMBER 1965

NASA TN D-3027

ILLUSTRATION OF DESIGN METHODS FOR POWER REDUCTION IN LOGICAL SYSTEMS

By H. Allen Curtis

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 – Price $1.00

ILLUSTRATION OF DESIGN METHODS FOR POWER REDUCTION IN LOGICAL SYSTEMS

by H. Allen Curtis

Lewis Research Center

SUMMARY

Power consumption in space communications and telemetry systems can be reduced considerably by the use of micropower transistor logic circuits and the incorporation of data compression techniques. In this report the additional importance of the conscientious use of systematic design procedures in the optimization of logic circuitry for such systems is emphasized.

The primary goal of this report is to bring to the attention of logic designers many of the useful systematic methods developed in recent years. The medium used to reach this goal is the presentation of a design problem - the derivation of a comparator unit for a telemetry data system. It is shown that a judicious use of ingenuity and systemmatic methods affords considerable savings in both components and power over previous designs obtained by ingenuity alone.

INTRODUCTION

Power requirements are critical in deep-space probes. This criticalness occurs because available solar power decreases as the distance from the Sun increases. It has been shown that a substantial reduction of power consumption for the logic used in aerospace telecommunications systems can be afforded by the use of micropower transistor logic circuits (ref. 1).

Usually in telecommunications systems continuous measurements are made, stored, and transmitted back to Earth. There is a considerable waste of power in storing and transmitting these data when they are highly redundant. Consequently, a further reduction of power consumption can be obtained whenever a major portion of the redundancy in the data is removed before transmission. Data compression has been offered as a means of achieving such a removal of data redundancy (refs. 2 and 3).

A third means of reducing power consumption is provided by the use of design procedures that tend to minimize the number of logical components necessary in telecommunications systems. This way, possibly because of its obviousness, has not been stressed enough. Therefore, the investigations of this report are concerned solely with the use of logical design methods for achieving reductions in power consumption.

In the summer of 1964, as a part of an investigation of micropower logic circuits, a data compressor was being designed to illustrate the use of recently developed micropower logic modules (ref. 4). A request that the author attempt to optimize the logical design of a portion of the compressor furnished the motivation for the study comprising this report.

It is hoped that through this report logic designers will be motivated to investigate more extensively the literature for new systematic procedures for solving their design problems. A few such procedures that are particularly suited to the solution of the illustrative problem are exploited in this report. Besides these procedures, which are cited in the reference section, a small but basic list of papers and books containing useful design techniques is included in the form of a bibliography. No attempt has been made to present rules for determining which methods are appropriate for any given problems. Such determinations can only be effectively made by the designer through his experience with the many methods. Investigations on iterative arrays of logical circuits lend support to the contention that there exists no general set of rules for the solution of any logic design problem (ref. 5). Example problems presented in the references and bibliography should give the designer an insight into the appropriateness of methods to his specific problems.

DESCRIPTION OF TYPICAL DATA COMPRESSOR

In the basic type of telemetry data compressor shown in figure 1 there are three major units - a predictor, a comparator, and a buffer. In accordance with past data samples, the predictor predicts the next data sample. The comparator then compares the latest sample with the predicted value to within an amplitude tolerance band defining the prediction error allowed for the accuracy specified by the data user. If the sample data point and the predicted point lie within the tolerance band, the sample is considered redundant and is consequently discarded. If, however, the data sample is outside the tolerance band of the predicted value, it is retained and sent along with a record of the sampling time to the buffer for later transmission over the data link.

The prediction criterion used in most compressors is a simple one. The predicted value is chosen to be equal to the last transmitted sample. Hence, when t, p, and s stand for the tolerance, the predicted point, and the sample point, respectively, the prediction criterion required for transmission is

$$t < |p - s|$$

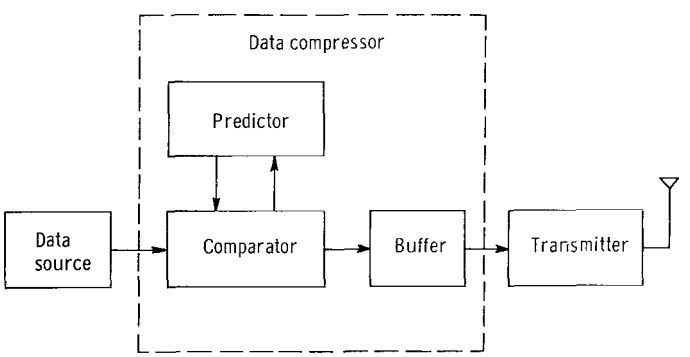


Figure 1. - Block diagram of a data compressor.

In a recent study of prediction criteria it was found that the more complex criteria did not tend to yield greater compression for any desired accuracy (ref. 2).

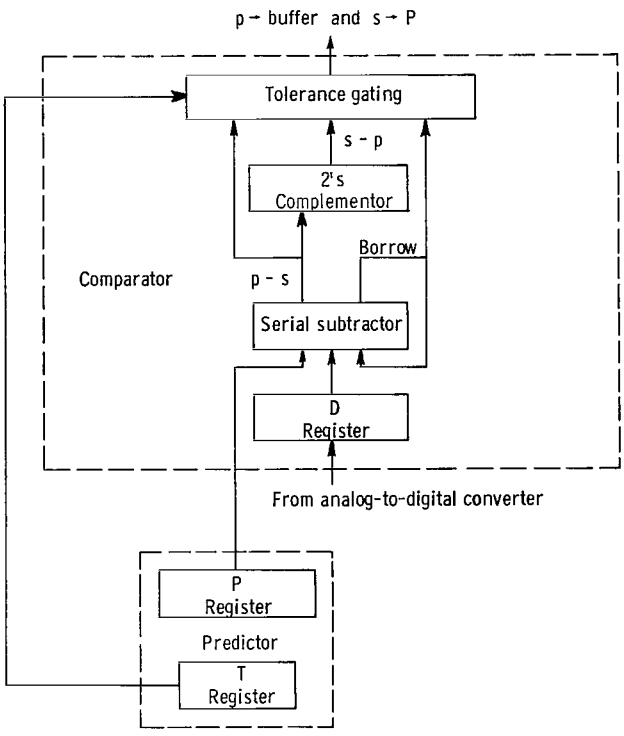


Figure 2. - Block diagram of typical predictor and comparator.

The simplicity of the criterion, $t < |p - s|$, is reflected in the fact that the associated predictor unit requires a minimum of circuitry. It consists merely of a memory register P containing the binary representation of the predicted value p and another register T containing the binary representation of the tolerance t. Typically, P and T would have seven and three bit capacities, respectively. Henceforth, they will be considered to have the aforementioned capacities. Except for the purpose of comparing various specific designs, the use of generalized capacities would not handicap ensuing discussions.

A comparator unit which has been suggested for such a data compressor (ref. 2) consists of a seven bit register D for the data sample s, a serial subtractor, a 2's complementor, and a tolerance gating network. This comparator, illustrated along with the predictor in figure 2, works as follows. Commencing with the low-order bit position of both P and D, the difference p - s is sequentially calculated by the serial subtractor. The 2's complementor derives the difference s - p from the output of the subtractor. If the final borrow eminating from the subtractor is a 0, then the difference p - s is nonnegative and hence equal to $|p - s|$. However, if this borrow is a 1, then the difference p - s is negative, and $s - p = |p - s|$. The quantities t and $|p - s|$ are compared in the tolerance gating network. If t is found to be less than $|p - s|$ at the end of the subtraction process, then the contents of P are sent to the buffer and the contents of D to P.

It should be noted in figure 2 that an analog-to-digital converter is the data source for the comparator. The converter, using the successive approximation technique (ref. 4), works on an eight-clock pulse cycle. The first seven pulses of the cycle convert bit by bit, high order first, the data sample s from its analog form to its binary representation. The eighth pulse is used to read the data sample s out of the converter memory register into the D register of the comparator. The comparator unit then uses the converter clock source to perform its functions while the next data sample is being converted.

DEFICIENCIES OF TYPICAL AND ALLIED ALTERNATIVE COMPARATOR DESIGNS

If the converter and the comparator were synchronized, it is conceivable that the data sample s could be taken directly from the converter memory in the comparison process and the D register could be eliminated. Synchronization

could be achieved by changing the analog-to-digital converter clock pulse cycle: The first seven pulses could take care of the analog-to-digital conversion of s. The converter could be idle during the time of the next seven pulses in which the comparator performs its duties. A final pulse time would be used to transfer p to the buffer and send the contents of the converter memory register to P. In order to retain the same sampling rate as before, the clock rate of the converter would have to be doubled. This, however, makes this synchronization scheme impractical since doubling the clock rate appreciably increases the power consumption.

Synchronization could also be achieved by designing an analog-to-digital converter to work from low order to high along with the comparator. However, the complexity of the "look-ahead" circuitry required to allow analog-to-digital conversion from low order to high is substantially greater than that of the D register, which could be saved. Thus, it is apparent that if synchronization is to be obtained, the comparator design rather than the converter design must be changed.

The typical comparator was designed by the use of ingenuity in the assembling of well-known units which are found in computers (serial subtractor, 2's complementor, and two number serial comparison unit). "Ingenuity" methods as opposed to systematic methods can similarly be used to design alternative comparators that do permit synchronization and hence do not require a D register. Two such comparators are described, and their deficiencies are noted.

The first comparator consists of a parallel subtractor in which the difference $t - |p - s|$ is calculated. This calculation is made entirely during the eighth pulse time of the converter cycle, when s is available in the converter memory. This design does not require a D register in the comparator and also eliminates the need for circuitry to gate serially the contents of P, D, and T registers. Each stage or position of such a parallel subtractor is of a complexity approximately equal to that of a serial subtractor. Therefore, the preferability of a parallel comparator over that of the serial comparator depends on the number of stages of P. For the typical seven stages, the parallel comparator unit, even with the aforementioned savings, is more complex than the serial comparator unit.

The second type of comparator is of the same serial design as the typical one of figure 2 but works on an eight pulse cycle that occurs entirely during the final pulse time of the analog-to-digital converter pulse cycle. Because the complete s information is available in the converter memory at this time, the D register is no longer necessary. The D register has been eliminated at the expense of adding a clock source to the comparator. Just as increasing the pulse rate of the analog-to-digital converter substantially increased power consumption, so does the increasing of the pulse rate to the comparator. Hence, this design offers no possibility for improvement.

ANALYSIS OF REQUIREMENTS FOR AN OPTIMIZED COMPARATOR

There have been developed in recent years many algorithms or systematic procedures for designing nearly minimal or minimal logical networks. These

methods for large systems, unless obviously decomposable into a set of simpler systems, are usually too unwieldy to be practicable. With the present state of the art, therefore, ingenuity methods are often the only means of designing large complex systems. Designers, because of working almost entirely with large systems, frequently overlook the possibility of applying methods other than those based entirely on ingenuity when faced with smaller sized system configurations.

The comparator unit definitely falls into the category of a small system. Yet, it appears that until now only ingenuity methods have been applied to comparator unit design. To apply one or more of the systematic procedures the logical specifications and requirements for the comparator must be converted from a verbal to a more tractable form.

This form should ideally be a mathematical model of the comparator. The model should define the action of the comparator, and through this definition should reveal the fundamental structure of the comparator. Such a model is obtained when one represents the comparator as a finite state sequential machine. This machine has a set of inputs, an initial state, ensuing states, and an output. The output merely permits the transfer of p to the buffer and of s from the converter register to P whenever the relation $t < |p - s|$ is satisfied. The inputs to the machine are the data from the T and P registers of the predictor and from the memory of the analog-to-digital converter; this latter memory will henceforth be referred to as the S register. For synchronization of the machine and the converter, these inputs are to be serially gated, bit position by bit position, high order first. Figure 3 provides a block diagram of the sequential machine description of the comparator.

Let t_k, p_k, and s_k represent the binary information contained in the kth position ($0 \leq k \leq 6$) of the T, P, and S registers, respectively. The four high-order positions of T do not exist. However, for uniformity it is convenient to consider T as a seven position register with t_3, t_4, t_5, and t_6 understood to be 0. There are clearly eight distinct sequences of inputs $\{t_k, p_k, s_k\}$ that could be gated into the machine: {0,0,0}, {0,0,1}, {0,1,0}, {0,1,1}, {1,0,0}, {1,0,1}, {1,1,0}, and {1,1,1}.

The binary representations of the tolerance, the predicted value, and the sample value are given by

$$t = \sum_{i=0}^{6} 2^i t_i$$

$$p = \sum_{i=0}^{6} 2^i p_i$$

$$s = \sum_{i=0}^{6} 2^i s_i$$

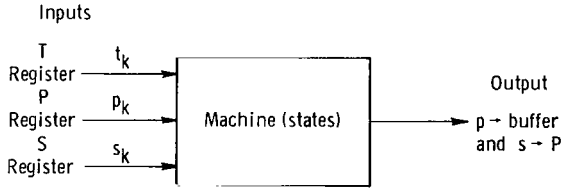


Figure 3. - Block diagram of comparator represented as a sequential machine.

The first pulse of each eight-clock pulse conversion cycle serves both to derive s_6 in S and to initialize the machine to its initial state. The second pulse, besides deriving s_5, gates the inputs $\{t_6, p_6, s_6\}$ into the machine which subsequently goes into a new state. Similarly, the $(8 - k)$th converter clock pulse gates the inputs $\{t_k, p_k, s_k\}$ into the machine, and the machine then assumes another new state.

The information that has thus far been gated into the machine is precisely that held in the high order $7 - k$ positions of T, P, and S. This information is given by the partial representations

$$\tau_k = \sum_{i=k}^{6} 2^i t_i$$

$$\pi_k = \sum_{i=k}^{6} 2^i p_i$$

$$\sigma_k = \sum_{i=k}^{6} 2^i s_i$$

from which it is evident that $\tau_0 = t$, $\pi_0 = p$, and $\sigma_0 = s$. The new state of the machine carries information about τ_k, π_k, and σ_k and about how they are related in regard to the eventual satisfaction of the relation $t < |p - s|$. After the eighth pulse the final state of the machine determines the output condition enabling or inhibiting the transmission of p to the buffer and s to P. The first pulse of the next cycle, in addition to performing the functions previously described, provides the time interval for this transmission if it is enabled.

To transform completely the verbal description of the machine into a mathematical model of the comparator, the precise nature of each of the possible states of the machine must be derived. To facilitate the derivation, three theorems concerning relations of τ_k, π_k, and σ_k to $t < |p - s|$ will now be presented. These relations are those which the states of the machine are to convey.

Theorem 1:

$$\text{If } \tau_k + 2^{k+1} - 2 < |\pi_k - \sigma_k|, \text{ then } t < |p - s|.$$

Proof:

For $k = 0$, the theorem is trivially true since

$$\tau_0 + 2 - 2 < |\pi_0 - \sigma_0|$$

is precisely

$$t < |p - s|$$

According to the hypothesis,

$$\tau_k + 2^{k+1} - 2 < |\pi_k - \sigma_k|$$

Then

$$\tau_k + 2^k - 1 < |\pi_k \quad \sigma_k| - 2^k + 1$$

Given τ_k, the maximum value that t can attain is $\tau_k + 2^k - 1$:

$$t = \sum_{i=0}^{6} 2^i t_i = \sum_{i=k}^{6} 2^i t_i + \sum_{i=0}^{k-1} 2^i t_i = \tau_k + \sum_{i=0}^{k-1} 2^i t_i$$

$$t_{max} = \tau_k + \sum_{i=0}^{k-1} 2^i = \tau_k + 2^k - 1$$

Likewise given $|\pi_k - \sigma_k|$, the minimum value that $|p - s|$ can attain is $|\pi_k - \sigma_k| - 2^k + 1$:

$$|p - s| = \left|\sum_{i=0}^{6} 2^i p_i - \sum_{i=0}^{6} 2^i s_i\right|$$

$$= |\sum_{i=k}^{6} 2^i p_i - \sum_{i=k}^{6} 2^i s_i + \sum_{i=0}^{k-1} 2^i (p_i - s_i)|$$

$$= |\pi_k - \sigma_k + \sum_{i=0}^{k-1} 2^i (p_i - s_i)|$$

$$|p - s|_{min} = |\pi_k - \sigma_k| - \sum_{i=0}^{k-1} 2^i = |\pi_k - \sigma_k| - 2^k + 1$$

Thus, $t_{max} < |p - s|_{min}$, and it follows that $t < |p - s|$.

Theorem 2:

If $\tau_k > |\pi_k - \sigma_k| + 2^k - 1$, then $t > |p - s|$.

Proof:

For $k = 0$, the theorem is trivially true because $\tau_0 > |\pi_0 - \sigma_0| + 1 - 1$ is exactly $t > |p - s|$.

By the hypothesis,

$$\tau_k > |\pi_k - \sigma_k| + 2^k - 1$$

Given τ_k, the minimum value that t can attain is τ_k since

$$t = \tau_k + \sum_{i=0}^{k-1} 2^i t_i$$

or

$$t_{min} = \tau_k$$

Similarly, given $|\pi_k - \sigma_k|$, the maximum value that $|p - s|$ can attain is $|\pi_k - \sigma_k| + 2^k - 1$ as follows:

$$|p - s| = |\pi_k - \sigma_k + \sum_{i=0}^{k-1} 2^i (p_i - s_i)|$$

$$|p - s|_{max} = |\pi_k - \sigma_k| + \sum_{i=0}^{k-1} 2^i = |\pi_k - \sigma_k| + 2^k - 1$$

Therefore, $t_{min} > |p - s|_{max}$; consequently, $t > |p - s|$.

THEOREM 3:

If $|\pi_k - \sigma_k| - 2^{k+1} + 2 \leq \pi_k \leq |\pi_k - \sigma_k| + 2^k - 1$,

then either $\tau_k = |\pi_k - \sigma_k|$ or else $\tau_k = |\pi_k - \sigma_k| - 2^k$.

PROOF:

It was seen that

$$\tau_k = \sum_{i=k}^{6} 2^i t_i = 2^k \sum_{i=k}^{6} 2^{i-k} t_i = 2^k \alpha$$

where α is a nonnegative integer. Similarly,

$$|\pi_k - \sigma_k| = 2^k \left| \sum_{i=k}^{6} 2^{i-k} (p_i - s_i) \right| = 2^k \beta$$

where β is a nonnegative integer. The hypothesis can be rewritten as

$$2^k \beta - 2^{k+1} + 2 \leq 2^k \alpha \leq 2^k \beta + 2^k - 1$$

The maximum integer α satisfying the hypothesis is clearly β, and the minimum integer α is $\beta - 1$. Therefore, there are only two values of α possible, and correspondingly, since $2^k \beta = |\pi_k - \sigma_k|$, $\tau_k = |\pi_k - \sigma_k|$ or $\tau_k = |\pi_k - \sigma_k| - 2^k$. For the case $k = 0$, it should be noted that only one value is possible: $t = |p - s|$.

Theorems 1 and 2 reveal the conditions necessary for the satisfaction of the relations $t < |p - s|$ and $t > |p - s|$, respectively. With each of these relations there is associated a distinct state of the machine. In particular, let the state characterized by the satisfaction of $t < |p - s|$ be referred to as state 1. Likewise, the state associated with $t > |p - s|$ will be called state 2. Theorem 3 provides information about additional relations that must be satisfied by the machine in its other possible states. Conclusions that can be drawn by examining the three theorems are the following:

(1) For any clock pulse cycle, once the machine has reached either state 1 or state 2, it cannot change to a different state regardless of further sequences of inputs.

(2) Since the initial state of the machine is the state occurring before any sequences of inputs have been gated into the machine, it can convey no information regarding whether or not $t < |p - s|$ or $t > |p - s|$. Therefore, the initial state must be one of those associated with the relations treated in theorem 3.

(3) If, after all of the input sequences have occurred, the machine is not in state 1 or state 2, then $t = |p - s|$.

The properties of the states associated with the relations $\tau_k = |\pi_k - \sigma_k|$ and $\tau_k = |\pi_k - \sigma_k| - 2^k$ are now investigated. The former relation represents a class containing three simpler relations:

$$\tau_k = \pi_k - \sigma_k = 0$$

$$\tau_k = \pi_k - \sigma_k > 0$$

$$\tau_k = \sigma_k - \pi_k > 0$$

Similarly, $\tau_k = |\pi_k - \sigma_k| - 2^k$ represents a class of two simpler relations:

$$\tau_k = \pi_k - \sigma_k - 2^k$$

$$\tau_k = \sigma_k - \pi_k - 2^k$$

Let the state of the machine for which the first of these five simple relations is satisfied be called state 3. Similarly, associate with the other four relations the states 4, 5, 6, and 7, respectively. Whether or not the machine can be described by fewer states will be determined later as part of the design procedure.

If the present relation and state of the machine are $\tau_k = \pi_k - \sigma_k = 0$ and 3, respectively, when the sequence $\{t_{k-1}, p_{k-1}, s_{k-1}\}$ is gated into the machine, then the next relation and state can be determined by the use of the recursion formulas:

$$\tau_{k-1} = \tau_k + 2^{k-1}t_{k-1}$$

$$\pi_{k-1} - \sigma_{k-1} = \pi_k - \sigma_k + 2^{k-1}(p_{k-1} - s_{k-1})$$

TABLE I. - PRESENT STATE 3 AND PRESENT RELATION

$\tau_k = \pi_k - \sigma_k = 0$

Input sequence			Next relation	Next state
t_{k-1}	p_{k-1}	s_{k-1}		
0	0	0	$\tau_{k-1} = \pi_{k-1} - \sigma_{k-1} = 0$	3
0	0	1	$\tau_{k-1} = \sigma_{k-1} - \pi_{k-1} - 2^{k-1}$	7
0	1	0	$\tau_{k-1} = \pi_{k-1} - \sigma_{k-1} - 2^{k-1}$	6
0	1	1	$\tau_{k-1} = \pi_{k-1} - \sigma_{k-1} = 0$	3
1	0	0	$\tau_{k-1} > \lvert\pi_{k-1} - \sigma_{k-1}\rvert + 2^{k-1} - 1$	2
1	0	1	$\tau_{k-1} = \sigma_{k-1} - \pi_{k-1} > 0$	5
1	1	0	$\tau_{k-1} = \pi_{k-1} - \sigma_{k-1} > 0$	4
1	1	1	$\tau_{k-1} > \lvert\pi_{k-1} - \sigma_{k-1}\rvert + 2^{k-1} - 1$	2

TABLE II. - PRESENT STATE 4 AND PRESENT RELATION

$\tau_k = \pi_k - \sigma_k > 0$

Input sequence			Next relation	Next state
t_{k-1}	p_{k-1}	s_{k-1}		
0	0	0	$\tau_{k-1} = \pi_{k-1} - \sigma_{k-1} > 0$	4
0	0	1	$\tau_{k-1} > \lvert\pi_{k-1} - \sigma_{k-1}\rvert + 2^{k-1} - 1$	2
0	1	0	$\tau_{k-1} = \pi_{k-1} - \sigma_{k-1} - 2^{k-1}$	6
0	1	1	$\tau_{k-1} = \pi_{k-1} - \sigma_{k-1} > 0$	4
1	0	0	$\tau_{k-1} > \lvert\pi_{k-1} - \sigma_{k-1}\rvert + 2^{k-1} - 1$	2
1	0	1	$\tau_{k-1} > \lvert\pi_{k-1} - \sigma_{k-1}\rvert + 2^{k-1} - 1$	2
1	1	0	$\tau_{k-1} = \pi_{k-1} - \sigma_{k-1} > 0$	4
1	1	1	$\tau_{k-1} > \lvert\pi_{k-1} - \sigma_{k-1}\rvert + 2^{k-1} - 1$	2

The result of such a determination is given in table I.

In a like manner tables relating successive states of the machine are derived for states 4, 5, 6, and 7. For completeness they are presented in tables II to V.

From these tables and the conclusions drawn from the three theorems another table, which relates present state, next state, and output when input sequences are applied to the machine, is directly derivable. If, at the time of the final pulse of the cycle, the machine is in state 1, 6, or 7, then the transfer of p to the buffer and of s to P is enabled. The enabling output condition of the machine is indicated in the table by the presence of a 1 in the output column. This table provides the sought for mathematical representation of the comparator and is a specific example of a flow table representation of a finite state sequential machine (refs. 6 and 7).

SYSTEMATIC DESIGN PROCEDURE

An important step in the design of sequential machines is the assignment of binary variables to represent the internal states of the machines. The problem of determining economical state assignments for sequential machines has been studied rather extensively (refs. 7 to 9). These investigations have been motivated by the fact that different assignments result in different logical relations with corresponding variations in the complexity of hardware implementations. Resulting from these investigations are methods that systematically produce assignments in which each binary variable describing the new state depends on as few variables of the old state as possible.

The first step in the systematic design of the comparator consists of applying the method of Dr. Juris Hartmanis (ref. 7). This method frequently will not yield variable assignments in which dependence is reduced to the greatest extent. However, the method has the advantage that its computation is slight

TABLE III. - PRESENT STATE 5 AND PRESENT RELATION

$\tau_k = \sigma_k = \pi_k > 0$

Input sequence			Next relation	Next state
t_{k-1}	p_{k-1}	s_{k-1}		
0	0	0	$\tau_{k-1} = \sigma_{k-1} - \pi_{k-1} > 0$	5
0	0	1	$\tau_{k-1} = \sigma_{k-1} - \pi_{k-1} - 2^{k-1}$	7
0	1	0	$\tau_{k-1} > \lvert\pi_{k-1} - \sigma_{k-1}\rvert + 2^{k-1} - 1$	2
0	1	1	$\tau_{k-1} = \sigma_{k-1} - \pi_{k-1} > 0$	5
1	0	0	$\tau_{k-1} > \lvert\pi_{k-1} - \sigma_{k-1}\rvert + 2^{k-1} - 1$	2
1	0	1	$\tau_{k-1} = \sigma_{k-1} - \pi_{k-1} > 0$	5
1	1	0	$\tau_{k-1} > \lvert\pi_{k-1} - \sigma_{k-1}\rvert + 2^{k-1} - 1$	2
1	1	1	$\tau_{k-1} > \lvert\pi_{k-1} - \sigma_{k-1}\rvert + 2^{k-1} - 1$	2

TABLE IV. - PRESENT STATE 6 AND PRESENT RELATION

$\tau_k = \pi_k - \sigma_k - 2^k$

Input sequence			Next relation	Next state
t_{k-1}	p_{k-1}	s_{k-1}		
0	0	0	$\tau_{k-1} < \lvert\pi_{k-1} - \sigma_{k-1}\rvert - 2^k + 2$	1
0	0	1	$\tau_{k-1} = \pi_{k-1} - \sigma_{k-1} - 2^{k-1}$	6
0	1	0	$\tau_{k-1} < \lvert\pi_{k-1} - \sigma_{k-1}\rvert - 2^k + 2$	1
0	1	1	$\tau_{k-1} < \lvert\pi_{k-1} - \sigma_{k-1}\rvert - 2^k + 2$	1
1	0	0	$\tau_{k-1} = \pi_{k-1} - \sigma_{k-1} - 2^{k-1}$	6
1	0	1	$\tau_{k-1} = \pi_{k-1} - \sigma_{k-1} > 0$	4
1	1	0	$\tau_{k-1} > \lvert\pi_{k-1} - \sigma_{k-1}\rvert - 2^k + 2$	1
1	1	1	$\tau_{k-1} = \pi_{k-1} - \sigma_{k-1} - 2^{k-1}$	6

for machines having few states and that it also determines what states, if any, are redundant. Furthermore, one can bypass much of the computation and still obtain valuable insight into the fundamental structure of the machine. This latter information can then be used to assess the merits of assignments obtained by more complex methods.

The execution of this method permits one to determine the existence of assignments of binary variables in which a subset of these variables can be calculated independently from the remaining variables. The existence of such assignments is closely connected with the existence of partitions with the substitution property. Since the partition with the substitution property is the main tool of the method, it would be well to provide a somewhat detailed definition of this entity. A block is a subset of the set of states of a sequential machine M. A partition is a union of blocks such that every state of M is included once and only once. A partition ρ on the set of states of a sequential machine M is said to have the substitution property with respect to M if, for any two states contained in the same block of ρ, their next states will again be contained in a common block as long as the same input was applied. (Hartmanis uses the symbol π instead of ρ. For purposes of avoiding ambiguity, the choice of ρ for this report is preferable.) Note that the partition $\rho_1 = \{\overline{1\ 2}\ \overline{3}\ \overline{4}\ \overline{5}\ \overline{6}\ \overline{7}\}$ has the substitution property with respect to the comparator as given by its flow table representation. Clearly, the states 1 and 2 contained in block $\overline{1\ 2}$ are again contained in $\overline{1\ 2}$ with the application of any input sequence; none of the other blocks contains more than one state. The computational scheme quickly yields the other partitions with the substitution property

$$\rho_2 = \{\overline{1\ 2\ 4\ 6}\ \overline{3}\ \overline{5}\ \overline{7}\}$$

$$\rho_3 = \{\overline{1\ 2\ 5\ 7}\ \overline{3}\ \overline{4}\ \overline{6}\}$$

TABLE V. - PRESENT STATE 7 AND PRESENT RELATION

$$\tau_k = \sigma_k - \pi_k - 2^k$$

Input sequence			Next relation	Next state
t_{k-1}	p_{k-1}	s_{k-1}		
0	0	0	$\tau_{k-1} < \lvert\pi_{k-1} - \sigma_{k-1}\rvert - 2^k + 2$	1
0	0	1	$\tau_{k-1} < \lvert\pi_{k-1} - \sigma_{k-1}\rvert - 2^k + 2$	1
0	1	0	$\tau_{k-1} = \sigma_{k-1} - \pi_{k-1} - 2^{k-1}$	7
0	1	1	$\tau_{k-1} < \lvert\pi_{k-1} - \sigma_{k-1}\rvert - 2^k + 2$	1
1	0	0	$\tau_{k-1} = \sigma_{k-1} - \pi_{k-1} - 2^{k-1}$	7
1	0	1	$\tau_{k-1} < \lvert\pi_{k-1} - \sigma_{k-1}\rvert - 2^k + 2$	1
1	1	0	$\tau_{k-1} = \sigma_{k-1} - \pi_{k-1} > 0$	5
1	1	1	$\tau_{k-1} = \sigma_{k-1} - \pi_{k-1} - 2^{k-1}$	7

TABLE VI. - FLOW TABLE REPRESENTATION OF COMPARATOR

Present state	Input								Output
	000	001	010	011	100	101	110	111	
	Next state								
1	1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2	0
3	3	7	6	3	2	5	4	2	0
4	4	2	6	4	2	2	4	2	0
5	5	7	2	5	2	5	2	2	0
6	1	6	1	1	6	4	1	6	1
7	1	1	7	1	7	1	5	7	1

$$\rho_4 = \{\overline{1\ 2\ 5\ 7}\ \overline{3\ 4}\ \overline{6}\}$$

$$\rho_5 = \{\overline{1\ 2\ 4\ 6}\ \overline{3\ 5}\ \overline{7}\}$$

$$\rho_6 = \{\overline{1\ 2\ 4\ 5\ 6\ 7}\ \overline{3}\}$$

The fact that no nontrivial block (more than one state) contained in any of these partitions is such that all its states have the same output means the comparator has no superfluous states. Therefore, the derived set of states is minimal.

A simple calculation, based on the properties of the six partitions with the substitution property, reveals that the best assignment obtainable by the method is one requiring four binary variables to represent each state. If f_j and F_j $(j = 0,1,2,3)$ are used to represent the old and new state variables, respectively, then the old and new state variables for the best such assignment are related functionally as follows:

$$F_0 = g_0(f_0,t_k,p_k,s_k)$$

$$F_1 = g_1(f_0,f_1,t_k,p_k,s_k)$$

$$F_2 = g_2(f_0,f_2,t_k,p_k,s_k)$$

$$F_3 = g_3(f_0,f_1,f_2,f_3,t_k,p_k,s_k)$$

Assignments that have no reduced dependence require only three binary variables to represent each state. For each of these assignments

$$F_0 = g_0(f_0,f_1,f_2,t_k,p_k,s_k)$$

$$F_1 = g_1(f_0,f_1,f_2,t_k,p_k,s_k)$$

$$F_2 = g_2(f_0,f_1,f_2,t_k,p_k,s_k)$$

The application of a cost estimate procedure to the two sets of switching

functions (ref. 10) shows that the latter assignment is preferable. Cost in this procedure is defined as the number of logical components required. The fact that there is a close relation between the number of components and the amount of power consumed justifies the use of such a procedure.

The design procedure in its third step consists of applying a more general method (ref. 8) to obtain economical state assignments. Using this method, one can detect for any machine M all assignments that have reduced state variable dependence. Partition pairs, generalizations of partitions with the substitution property, are the principal instruments used in this method to discover assignments with reduced dependency. A partition pair $(\rho,\tilde{\rho})$ on the states of a sequential machine M is an ordered pair of partitions on the set of states such that if states S_i and S_j belong to the same block of ρ, then for each input sequence I, IS_i and IS_j are in the same block of $\tilde{\rho}$ (IS_i is the state the machine goes into from S_i when the input I is applied). When $\rho = \tilde{\rho}$, the definition of a partition pair reduces to that of a partition with the substitution property. To determine all partition pairs for the comparator from its flow table is relatively easy, although naturally more difficult than merely finding the partitions with the substitution property. In the set of all such partition pairs there are four that imply assignments which have reduced dependence and require only three variables to represent each state. These four pairs are as follows:

$$(\rho_7 = \{\overline{1\ 6}\ \overline{2\ 4}\ \overline{3\ 5}\ \overline{7}\},\ \tilde{\rho}_7 = \{\overline{1\ 2\ 4\ 6}\ \overline{3\ 5\ 7}\})$$

$$(\rho_8 = \{\overline{1\ 7}\ \overline{2\ 5}\ \overline{3\ 4}\ \overline{6}\},\ \tilde{\rho}_8 = \{\overline{1\ 2\ 5\ 7}\ \overline{3\ 4\ 6}\})$$

$$(\rho_9 = \{\overline{1\ 6}\ \overline{2\ 5}\ \overline{3\ 4}\ \overline{7}\},\ \tilde{\rho}_9 = \{\overline{1\ 3\ 4\ 6}\ \overline{2\ 5\ 7}\})$$

$$(\rho_{10} = \{\overline{1\ 7}\ \overline{2\ 4}\ \overline{3\ 5}\ \overline{6}\},\ \tilde{\rho}_{10} = \{\overline{1\ 3\ 5\ 7}\ \overline{2\ 4\ 6}\})$$

Each of these pairs implies the existence of an assignment for which

$$F_0 = g_0(f_0,f_1,t_k,p_k,s_k)$$

$$F_1 = g_1(f_0,f_1,f_2,t_k,p_k,s_k)$$

$$F_2 = g_2(f_0,f_1,f_2,t_k,p_k,s_k)$$

Moreover, each of two sets of pairs $(\rho_7, \tilde{\rho}_7)$, $(\rho_8, \tilde{\rho}_8)$ and $(\rho_9, \tilde{\rho}_9)$, $(\rho_{10}, \tilde{\rho}_{10})$ implies that

$$F_0 = g_0(f_0,f_2,t_k,p_k,s_k)$$

$$F_1 = g_1(f_1,f_2,t_k,p_k,s_k)$$

$$F_2 = g_2(f_0,f_1,f_2,t_k,p_k,s_k)$$

Such assignments have the sought after maximum reduction of dependency among the new and old state variables.

The assignment corresponding to the first set of two partition pairs is constructed as follows: Associate with the first variable (f_0 for the old and F_0 for the new) of each state the partition $\tilde{\rho}_7$ by letting this variable be 1 for each state in the $\underline{1\ 2\ 4\ 6}$ block and letting it be 0 for each of the states in $\overline{3\ 5\ 7}$. In a similar way the second variable, f_1 or F_1, of each state is associated with the partition $\tilde{\rho}_8$. The values of the third variable, f_2 or F_2, are obtained by associating $\tilde{\rho}_7$ with both the first and third variable and by associating ρ_8 with both second and third variables. The resulting assignment in tabular form is as follows:

	$f_2\ f_1\ f_0$		$F_2\ F_1\ F_0$
Present state	$1 \rightarrow 1\ 1\ 1$	Next state	$1 \rightarrow 1\ 1\ 1$
	$2 \rightarrow 0\ 1\ 1$		$2 \rightarrow 0\ 1\ 1$
	$3 \rightarrow 0\ 0\ 0$		$3 \rightarrow 0\ 0\ 0$
	$4 \rightarrow 0\ 0\ 1$		$4 \rightarrow 0\ 0\ 1$
	$5 \rightarrow 0\ 1\ 0$		$5 \rightarrow 0\ 1\ 0$
	$6 \rightarrow 1\ 0\ 1$		$6 \rightarrow 1\ 0\ 1$
	$7 \rightarrow 1\ 1\ 0$		$7 \rightarrow 1\ 1\ 0$

Substitution of these binary variables in the flow table representation of the comparator yields a truth table for the binary functions F_0, F_1, and F_2.

The final step of the design procedure is accomplished when the functions F_0, F_1, and F_2 are simplified in their logical forms. Before the last step is begun, it would be well to mention what kinds of logical switching elements are to be used in the hardware realization of the comparator. All these elements, to conserve power, are micropower transistor circuits. The memory elements are reset-set flip-flops (ref. 11). The basic logic elements are NAND, NOR, and EXCLUSIVE NOR. The NAND and NOR elements can have one to five inputs, whereas the EXCLUSIVE NOR element always has no more than two inputs. In figure 4 are shown the symbolic representations of the four aforementioned circuits. To each of these circuits is assigned a value reflecting the amount of power required to operate it (ref. 1). Each flip-flop or EXCLUSIVE NOR element is given a value of 2, but the NAND and NOR elements, regardless of the number of inputs, have a value of 1 each.

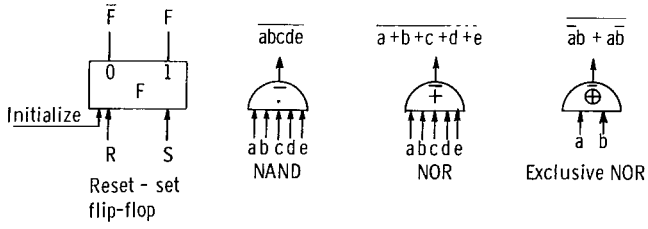


Figure 4. - Symbolic representation of micropower logic elements.

The flip-flop characteristic equation is

$$F = S + \bar{R} \cdot f$$

where R and S are the reset and set inputs to the flip-flop, f is the old state, F is the new state, + is the OR operation, · is the AND operation, and $\bar{}$ is the complementation operation. Since

$$F_0 = g_0(f_0, f_2, t_k, p_k, s_k)$$

it follows that

$$F_0 = S_0(f_0, f_2, t_k, p_k, s_k) + \bar{R}_0(f_0, f_2, t_k, p_k, s_k) \cdot f_0$$

From the truth table for F_0 it is a simple matter to determine the functions S_0 and R_0 (ref. 11). Because of the many "don't care" conditions for S_0 and R_0, they can both be selected to be independent of f_0. Once this selection is made, then

$$S_0 = \bar{f}_2\bar{t}_k p_k \bar{s}_k + \bar{f}_2 t_k \bar{p}_k \bar{s}_k + \bar{f}_2 t_k p_k \bar{s}_k + \bar{f}_2 t_k p_k s_k$$
$$+ f_2\bar{t}_k\bar{p}_k\bar{s}_k + f_2\bar{t}_k\bar{p}_k s_k + f_2\bar{t}_k p_k s_k + f_2 t_k \bar{p}_k s_k$$

$$R_0 = 0$$

The use of decomposition techniques (ref. 10) provides the final simplified form of S_0:

$$S_0 = f_2 \oplus [p_k\bar{s}_k + t(\bar{p}_k \oplus s_k)]$$

where $\oplus$ is the EXCLUSIVE OR operation. Similarly, the set and reset functions for F_1 and F_2 are found to be

$$S_1 = f_2 \oplus [\bar{p}_k s_k + t(\bar{p}_k + s_k)]$$

$$R_1 = 0$$

$$S_2 = \bar{f}_0\bar{t}_k\bar{p}_k s_k + \bar{f}_1\bar{t}_k p_k \bar{s}_k$$

$$R_2 = f_0\bar{f}_1 t_k \bar{p}_k s_k + \bar{f}_0 f_1 t_k p_k \bar{s}_k$$

The hardware realization of the comparator, corresponding to these logical relations, is shown in figure 5. Tabulated at the right of the circuit are the power consumption values. The total power consumption value of the optimized comparator is 23. To indicate the improvements afforded by the use of systematic design methods combined with ingenuity, the hardware realization of the typical comparator unit is shown in figure 6 for comparison. To obtain a power consumption value as low as 36 required the restriction that the tolerance could only assume certain values; the permitted values essentially specify the number of bits of precision desired. Even with this restriction for purposes of circuit simplification the typical comparator requires about 60 percent more power than does the optimized comparator. Moreover, if the registers P, D, and S had been given larger capacities, the savings in power would have been even more striking. For instance, if each of the aforementioned registers had had twelve stages instead of seven, the power consumption value of the typical comparator would have grown to 46 while the value associated with the optimized comparator would have remained fixed at 23.

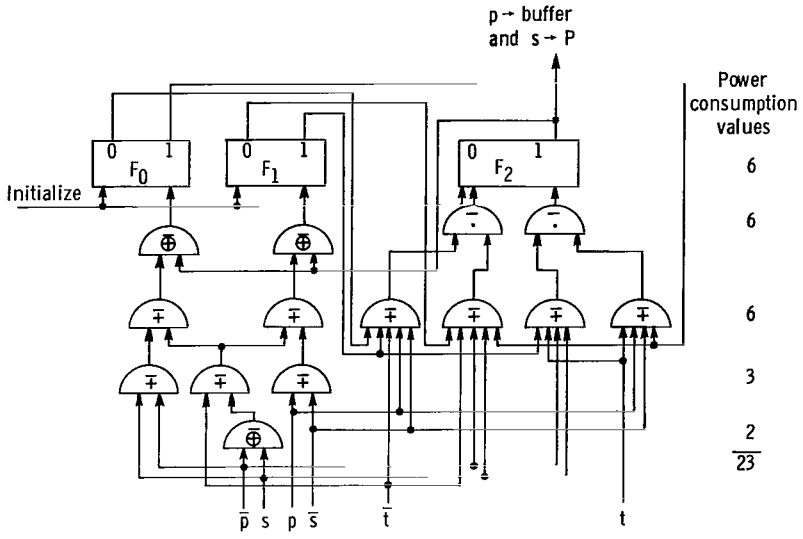


Figure 5. - Hardware realization of optimized comparator.

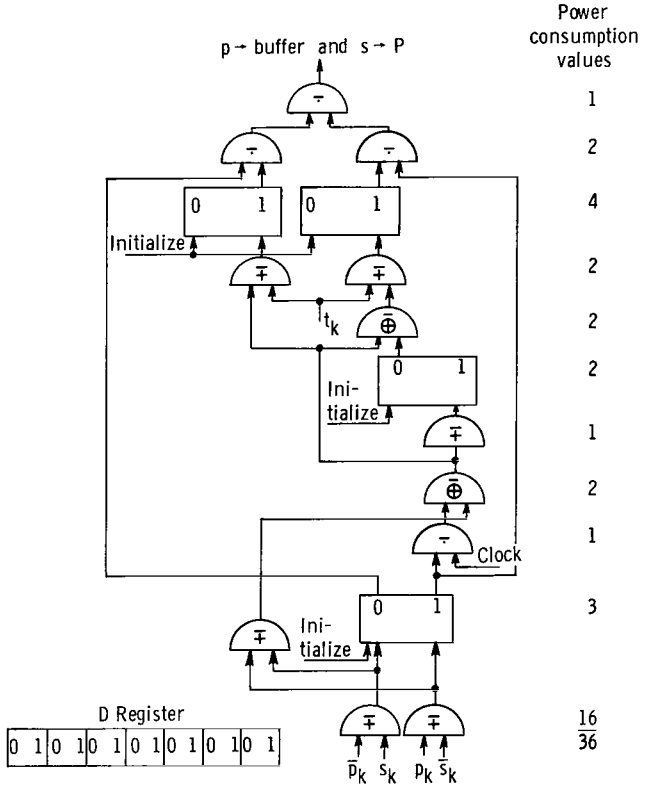


Figure 6. - Hardware realization of typical comparator.

If one were to execute the final step of the design procedure after making the assignment of state variables based on the second set of partition pairs, he would obtain a comparator whose power consumption value is 24. Therefore, the assignment which was chosen for illustration does indeed lead to the better hardware realization.

It is interesting to note that the application of the recently published method of Drs. T. A. Dolotta and E. J. McCluskey (ref. 12) to the comparator design problem yields precisely the circuit illustrated in figure 5. Their methods are of especial importance because they can be applied to much larger systems than could methods previously devised. For small systems this method does have one definite drawback. Assignments obtained by this method correspond to partitions chosen serially. Each partition is locally the best at each step of the process. However, sets of partitions chosen to interact the best globally will always provide the best assignment. Thus, assignments based on a serial selection of partitions will only be the best when they coincidentally interact the best globally.

CONCLUDING REMARKS

By judiciously combining ingenuity and the use of systematic design procedures a comparator whose power consumption is considerably less than any previously built has been derived. This comparator has an added advantage in that it permits a greater binary resolution of tolerances specified by the user. This optimized comparator has the additional feature that its power consumption does not increase for data samples of greater precision. These results, however, are of secondary importance compared to that of bringing to the attention of logic designers the many systematic design procedures that have been developed in recent years.

Lewis Research Center,
National Aeronautics and Space Administration,
Cleveland, Ohio, July 13, 1965.

REFERENCES

1. Sturman, John C.: Micropower Transistor Logic Circuits. NASA TN D-1462, 1963.

2. Medlin, J. E.: The Comparative Effectiveness of Several Telemetry Data Compression Techniques. Proc. Int. Telemetering Conf., London (England), Sept. 23-27, 1963. Vol. I. Inst. Elect. Engrs., 1963, pp. 328-340.

3. Blasbalg, H.; and Van Blerkom, R.: Message Compression. IRE Trans. on Space Electronics and Telemetry, vol. SET-8, no. 3, Sept. 1962, pp. 228-238.

4. Sturman, John C.; and Bertolino, Anthony V.: Low-Power, Low-Level Analog-to-Digital Converter for Space Vehicle Applications. NASA TN D-2916, 1965.

5. Hennie, F. C.: Iterative Arrays of Logical Circuits. John Wiley & Sons, Inc., 1961.

6. Huffman, D. A.: The Synthesis of Sequential Switching Circuits. J. Franklin Inst., vol. 257, no. 3, Mar. 1954, pp. 161-190, no. 4, Apr. 1954, pp. 275-303.

7. Hartmanis, J.: On the State Assignment Problem for Sequential Machines, I. IRE Trans. on Electronic Computers, vol. EC-10, no. 2, June 1961, pp. 157-165.

8. Curtis, H. A.: Multiple Reduction of Variable Dependency of Sequential Machines. J. Assoc. Comput. Machinery, vol. 9, no. 3, July 1962, pp. 324-344.

9. Stearns, R. E.; and Hartmanis, J.: On the State Assignment Problem for Sequential Machines, II. IRE Trans. on Electronic Computers, vol. EC-10, no. 4, Dec. 1961, pp. 593-603.

10. Curtis, H. A.: A New Approach to the Design of Switching Circuits. D. Van Nostrand, 1962.

11. Phister, M.: Logical Design of Digital Computers. John Wiley & Sons, Inc., 1958.

12. Dolotta, T. A.; and McCluskey, E. J.: The Coding of Internal States of Sequential Circuits. IEEE Trans. on Electronic Computers, vol. EC-13, no. 5, Oct. 1964, pp. 549-562.

BIBLIOGRAPHY

Caldwell, S. H.: Switching Circuits and Logical Design. John Wiley & Sons, Inc., 1958.

Curtis, H. A.: Generalized Tree Circuit - The Basic Building Block of an Extended Decomposition Theory. J. Assoc. Comput. Mach., vol. 10, no. 4, Oct. 1963, pp. 562-581.

Hartmanis, J.: Loop-Free Structure of Sequential Machines. Information and Control, vol. 5, no. 1, Mar. 1962, pp. 25-43.

Karp, R. M.: Some Techniques of State Assignment for Synchronous Sequential Machines. IEEE Trans. on Electronic Computers, vol. EC-13, no. 5, Oct. 1964, pp. 507-518.

Kohavi, Zvi: Secondary State Assignment for Sequential Machines. IEEE Trans. on Electronic Computers, vol. EC-13, no. 3, June 1964, pp. 193-203.

Lawler, E. L.: An Approach to Multilevel Boolean Minimization. J. Assoc. Comput. Mach., vol. 11, July 1964, pp. 283-295.

McCluskey, E. J., Jr.: Minimization of Boolean Functions. Bell System Tech. J., vol. 35, no. 6, Nov. 1956, pp. 1417-1444.

3/18/85

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Technical information generated in connection with a NASA contract or grant and released under NASA auspices.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

TECHNICAL REPRINTS: Information derived from NASA activities and initially published in the form of journal articles.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities but not necessarily reporting the results of individual NASA-programmed scientific efforts. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546